DTIC FILE COPY

SECURITY CLASSIFICATION OF THIS PAGE

(2)

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release, distribution unlimited
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR- 87-1646
6a. NAME OF PERFORMING ORGANIZATION Honeywell INc Physical Sciences Center 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION AFOSR/NE
6c. ADDRESS (City, State, and ZIP Code) 10701 Lyndale Avenue South Bloomington, MN 55420	7b. ADDRESS (City, State, and ZIP Code) Bldg 410 Bolling AFB, DC 20332-6448
8a. NAME OF FUNDING / SPONSORING ORGANIZATION SAME AS 7a 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F49620-86-C-0082
8c. ADDRESS (City, State, and ZIP Code) SAME AS 7b	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 61102F; PROJECT NO. DARPA; TASK NO. 5794/00; WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Optical Symbolic Processor for Expert System Execution

12. PERSONAL AUTHOR(S)
Derstine

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Quarterly Report	FROM 01Jun87 TO 31 Aug 87		

16. SUPPLEMENTARY NOTATION

DTIC ELECTE NOV 18 1987 S D CD

COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)

AD-A187 494

ABSTRACT (Continue on reverse if necessary and identify by block number)

The goal of this program is to develop a concept for an optical computer architecture for symbolic computing by defining a computation model of a high level language, examining the possible devices for the ultimate construction of a processor, and by defining required optical operations. In the quarter we undertook a detailed evaluation of our optical architecture (SPARO) for combinator graph reduction. Since we had determined that the interconnection network was the bottleneck in the performance of the architecture, our focus was on the message throughput of the simple register-based network. We derived an accurate performance model for the equivalent bidirectional ring network and found that the net parallelism in the architecture was indeed restricted by the low message traffice in the network. When messages exhibit no locality, the throughput for a 1024 processor network is limited to 8. With local messages, the maximum throughput for the same network is 27. All results were subsequently verified by simulations.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☐ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL DR C LEE GILES	22b. TELEPHONE (Include Area Code) (202) 767-4931 22c. OFFICE SYMBOL NE

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

AFOSR-TR- 87-1646

OPTICAL SYMBOLIC PROCESSOR FOR EXPERT SYSTEM EXECUTION

QUARTERLY TECHNICAL REPORT

June 1, 1987 to August 31, 1987

Sponsored by
Air Force Office of Scientific Research
and
Advanced Research Projects Agency (DOD)
ARPA Order No. 5794
Contract #F49620-86-C-0082

Prepared by

Aloke Guha
Subra Natarajan
Matthew Derstine

Honeywell Corporate Systems Development Division
Honeywell Physical Sciences Center

Submitted by: Aloke Guha
Aloke Guha, Principal Investigator

Approved by:
Anis Husain, Section Head

Approved by: D Fulkerson
David Fulkerson, Department Manager

SUMMARY

The goal of this program is to develop a concept for an optical computer architecture for symbolic computing by defining a computation model of a high level language, examining the possible devices for the ultimate construction of a processor, and by defining required optical operations.

In this quarter we undertook a detailed evaluation of our optical architecture (SPARO) for combinator graph reduction. Since we had determined that the interconnection network was the bottleneck in the performance of the architecture, our focus was on the message throughput of the simple register-based network. We derived an accurate performance model for the equivalent bidirectional ring network and found that the net parallelism in the architecture was indeed restricted by the low message traffic in the network. When messages exhibit no locality, the throughput for a 1024 processor network is limited to 8. With local messages, the maximum throughput for the same network is 27. All results were subsequently verified by simulations.

The poor performance of the simple ring network motivated us to examine other more elaborate but efficient interconnection network topologies. The alternatives considered included hybercubes, multistage interconnection networks (MINs) such as delta networks, and the single-stage shuffle-exchange amd replicated single-stage shuffle-exchange networks (SENs) where a number of SENs are used in parallel. On the basis of analysis and extensive simulations, we found SENs, especially replicated SENs, to be the most feasible and promising. Recent investigations have indicated that SENs could be implemented efficiently in optics. Furthermore, we have established that replicated SENs can provide a high throughput competitive with any other interconnection network.

The tasks forthcoming in the next quarter will focus on the final phase of the architecture design. We will reevaluate the modified optical architecture that uses a shuffle-exchange network, and tune the final architecture to obtain the maximum performance. This analysis will also yield the performance of the architecture in absolute terms, of reductions per second, for comparison with other electronic computers. We also plan to examine the detailed issues in implementing the architecture in optics and/or in electro-optics.

ANALYSIS OF INTERCONNECTION NETWORKS FOR SPARO

Our initial analysis focusses on the performance of the ring network that was proposed earlier. The performance metrics, both throughput and waiting time of messages are derived analytically, and compared with simulated results. The other candidate interconnection network, the shuffle exchange network, which has been found suitable for optical implementation, are also analysed. The results presented for the candidate networks are mostly from simulations. We show how shuffle-exchange networks, especially replicated shuffle exchange networks, can provide significant improvement in the message throughput and thus guarantee a greatly improved performance for SPARO.

1. INTRODUCTION

The performance of the proposed optical architecture, SPARO (Symbolic Processing Architecture in Optics), was shown to be dominated by the performance of the interconnection network. In order to determine the expected throughput of the messages in the SPARO network, and thus the rate of reductions, it is necessary to analyse the network used. A shift register based ring network was proposed in the original architecture. It was expected that the systolic nature of the ring network would accomplish a high throughput of messages, and therefore provide fast execution of combinator graphs by using a high degree of parallelism. Our analytical model reveals, however, that even a bidirectional ring network of large sizes cannot provide significant parallelism. Thus, while the simple ring network is amenable for optical implementation, its performance is not acceptable. This motivated us to examine alternative candidates for the network. Among the networks found suitable for packet switching, the single-stage (perfect) shuffle exchange network (SEN) and the binary hypercube appeared as likely candidates. We have therefore examined and analysed the SEN, and compared its performance against that of the hypercube and also the (electronically) popular multistage interconnection network (MIN), the delta network. The reason for using the delta network is that it is considered a standard high-performance interconnection topology. Unfortunately, the implementation of a MIN is much more complex than the SEN. Our intent in the comparisons was to show that despite the simpler implementation of the SEN in optics, (or in opto-electronics) the performance of the SEN is quite competitive with a MIN.

In the next section we have provided the detailed analytical modeling for the bidirectional ring network. We derive the expressions for both the throughput and the waiting time. Two cases are considered in our analysis of the ring network performance: i) the case where messages for any processor node are equally probable, i.e., no locality is assumed, and ii) locality of messages are assumed whereby the probability of the message to a destination node varies inversely as the distance from the source.

Section III examines the modeling for the single stage SEN. Our analysis is based on the work presented earlier by Lawrie and Padua [5]. It is assumed that the conflicts in the SEN are resolved by giving priority to the message closest to its destination. We show, from the results of our simulations, that for a packet switching network, even a modest message generation can throttle the network. This underscores the inappropriateness of the loading factor used by Lawrie and Padua to characterize the networks. To alleviate the problem of increased waiting times and low throughput, we studied buffered SENs. Contrary to naive expectations, the introduction of buffers do not improve performance. We present arguments as to why such behavior is observed, since

analytical modeling of buffered SENs with priority strategy for resolving conflicts is extremely difficult.

We end Section III by presenting our results on replicated SENs. We show how the performance of replicated networks can dramatically improve throughput. Based on our simulation results, we show what order replication would be recommended, given performance and cost constraints. The other alternative to replication is to use an enlarged network or a network that is about four times as large as the number of processors to be connected. The choice of replication or enlarging the network would be determined by the relative difficulty of merging multiple networks (in the case of replication) and the maximum size of the network that can be implemented (in the case of enlarging the network).

In Section IV, we present a summary on the performance analyses of hypercubes and delta networks for comparison with the SENs. The comparison is based on our simulated network results as well as the theoretical work done by other researchers.

Section V presents a brief note on the optical implementation of the SEN. We conclude in Section VI commenting on the possible implementation of the entire architecture for SPARO.

2. PERFORMANCE OF RING NETWORKS

Before presenting the model used to represent the ring network, we present below the assumptions made in our analysis. We also preface our assumptions with a brief description of the network.

2.1. Principle of operation

The register-based network originally designed for SPARO, purely by serendipity, looks quite similar to that proposed earlier for the ZMOB parallel processor [3] intended for image processing applications. The SPARO network is composed of at least 1024 registers (the size being determined by the size of problems that the architecture is targeted for) connected in a conveyor belt fashion. Each stage or register is associated at any time with a single processor node as in Figure 2.1. There are thus 1024 processor nodes. Each processor node can receive or send a message by accepting a message from or loading into its associated register in the network. Messages are delivered by the network by shifting the registers in a conveyor belt fashion. Since each message has a destination address, the message reaches its destination when the processor node address matches that of the message. Unlike the ZMOB network, the SPARO network is bidirectional. The network is assumed to recognize the direction which results in a shorter delivery path for a message. The analysis of unidirectional and bidirectional network is only trivially different.

In terms of operation, the following sequence is followed in SPARO:

i) Each processor in the network examines its buffer to see if the previous message has been delivered. If the buffer is full, the processor cannot load its new message into the buffer and will enter a wait state. Otherwise, the processor will load the buffer with its message and proceed with its processing.

ii) The ring network shifts and the ring register associated with each processor examines if it has a message to deliver to the processor. If so, the message is delivered.

iii) The buffer of each processor will check if the associated ring register is full (the message in the register is meant for another processor). If the register is empty,

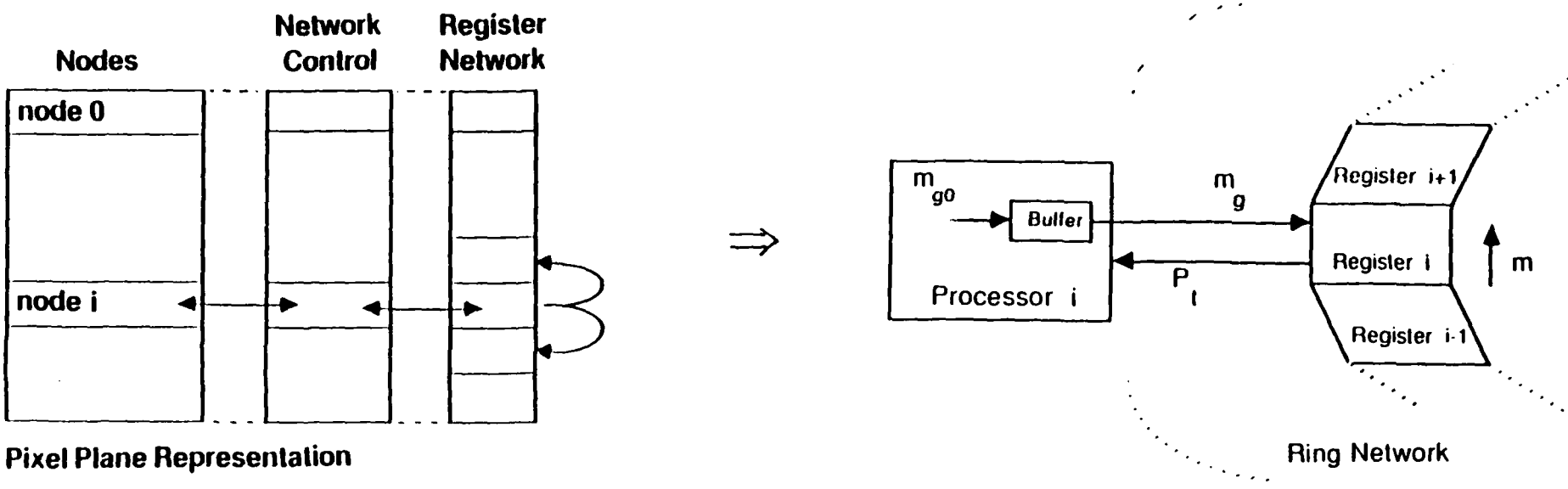


Figure 2.1 Modeling the SPARO ring network

then the buffer loads its message into the ring register. Otherwise, it waits.

Since in step (iii) the processor buffer cannot be emptied, the processor cannot generate more messages. This allows the ring to proceed uninterrupted at its full speed, and also ensures that no messages are lost. This assumption also implies that the message generation rate is influenced by the loading of the network. (The assumption reflects, especially in the case of fine-grained processing, that a processor operates on simple sequential tasks and cannot proceed until the previously sent message has been delivered and a response message has been received.) We now examine the analytical model of the ring network in brief.

2.2. Analytical model of the ring network

Figure 2.1 shows the representation of an individual stage of the network and its communication with the processors. We define below the following parameters that are used to define and compute the throughput of the network.

N=2n: total number of nodes in the network

p(i): probability that the destination of a message is i shifts away from source

m_{g0}: rate of message generation at each node under no loading restrictions

m_g: effective rate of message generation at each node

m: total rate at which messages arrive at a nodes via the network

P_t: probability of termination of a message arriving at a node

In the first part of our analysis, we consider the case that messages in the network are equally likely to be delivered to any node in the network. This contrasts with the case that the messages exhibit locality, or that the probability of accessing remote destinations is lower than those nearer to the source node. In the equiprobable case, the probability of generating a message with a destination that is i shifts away is independent of i. This probability is 2/N for a bidirectional ring (1/N for a unidirectional ring) where N=2n is the ring size.

We can calculate the rate m at which messages are traversing the network. If m_g be the effective probability of message generation in each processor, and P_t the probability that a message has terminated or reached its destination, then in the steady state, the rate of generation of messages, m_g, will be equal to the rate of consumption mP_t. Then, in a bidirectional ring,

$$m=m_g/2P_t$$

In the case of the unidirectional ring, m is twice that of the bidirectional ring.

Note that the above expression involves the effective message generation rate m_g and not the actual message generation rate which we denote by m_{g0}. This modification reflects the fact that the effective message generation rate depends on the load on the network. The effective message generation rate can be calculated by knowing when the buffer in the processor is full. If q_1 and q_0 be the probability that the buffer is full and empty, respectively, then we can find q_1 using the relation

$$q_1=a_{01}q_0+a_{11}q_1$$

where a_{01} and a_{11} are the probabilities of transition from q_0 to q_1 and vice versa. a_{01} is

thus the message generation rate when the network register is occupied, or $m_{g0}m(1-P_t)$. a_{11} is the probability that a non-terminating message arrives at the processor node or $m(1-P_t)$. Since $q_0=1-q_1$, we can show q_0 to be

$$q_0=\frac{1-m(1-P_t)}{1-m(1-P_t)(1-m_{g0})}$$

Using the expression for m previously derived, we find that m is the solution to the following equation.

$$m^2[2P_t(1-P_t)(1-m_{g0})]-m[2P_t-m_{g0}(1-P_t)]+m_{g0}=0$$

in the case of the unidirectional ring, the corresponding equation for m can be obtained by removing all occurrences of 2 from the above equation.

To calculate the throughput, we need to determine P_t, the probability of termination of any message. P_t in turn can be computed if the distribution of messages in steady state is known. By steady state distribution we mean the distribution probability of messages with different destination distances, from 1 to n (N for the unidirectional case).

To find the termination probability P_t, we first derive the distribution probability for messages with random destinations. Suppose two shifts have taken place in the network. The distribution of messages can be derived as follows. The number of messages that require $(i-1)$, $i\leq n$, shifts to reach its destination is 2+2. (The first term corresponds to the number of messages generated in the second shift that require $i-1$ shifts, while the second term corresponds to those that need the same number of shifts but are generated in the first shift.) Note that the 2 appearing in each term, except for the destination n away from the source, is due to the fact that we are considering bidirectional networks. Since the maximum number of shifts required for the ring network is n, the distribution of messages with different required shifts reaches steady state after n shifts. The number of messages requiring $i-1$ shifts is $2(n-i)+1$.

To derive the normalized probability, we need to find A where

$$A=\sum_{i=1}^{i=n} 2(n-i)+1=n^2$$

Thus, the probability that a message requires $i-1$ shifts to reach its destination:

$$p(i-1)=\frac{2(n-i)+1}{n^2}$$

The termination probability P_t is the probability that the destination of the message is 0 shifts away. This can be found by simply setting $i=1$ in the expression for $p(i-1)$, which gives $P_t=(2n-1)/n^2$. The termination probability for the unidirectional case can be shown to be $1/n$.

When messages exhibit locality, the same analysis can be carried out, except that the probability of messages requiring i shifts is inversely proportional to i (harmonic distribution). In such a case, P_t can be shown to be

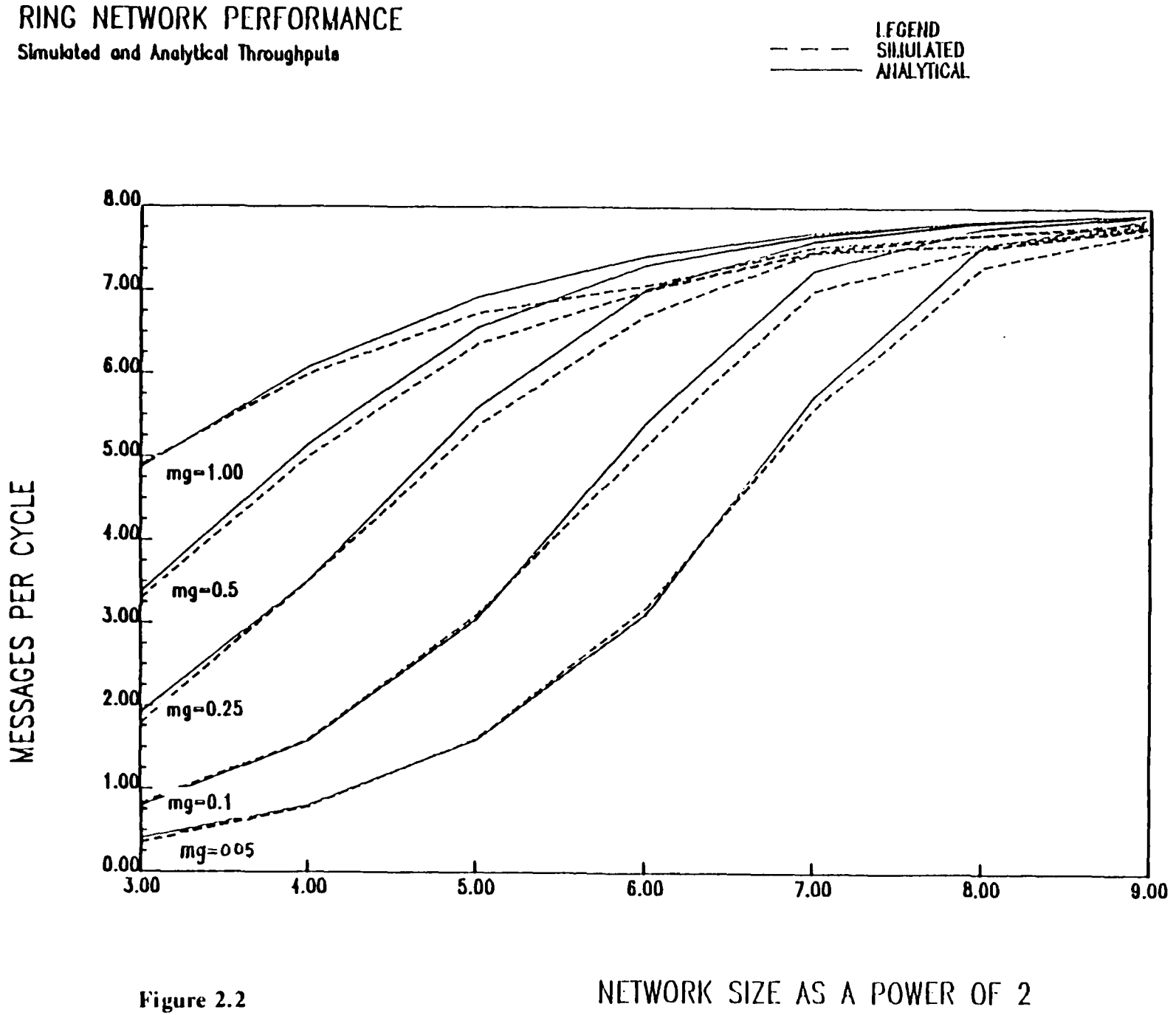

RING NETWORK PERFORMANCE
Simulated and Analytical Throughputs
LEGEND
SIMULATED
ANALYTICAL
MESSAGES PER CYCLE
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
0.00
mg=1.00
mg=0.5
mg=0.25
mg=0.1
mg=005
3.00
4.00
5.00
6.00
7.00
8.00
9.00
NETWORK SIZE AS A POWER OF 2

Figure 2.2

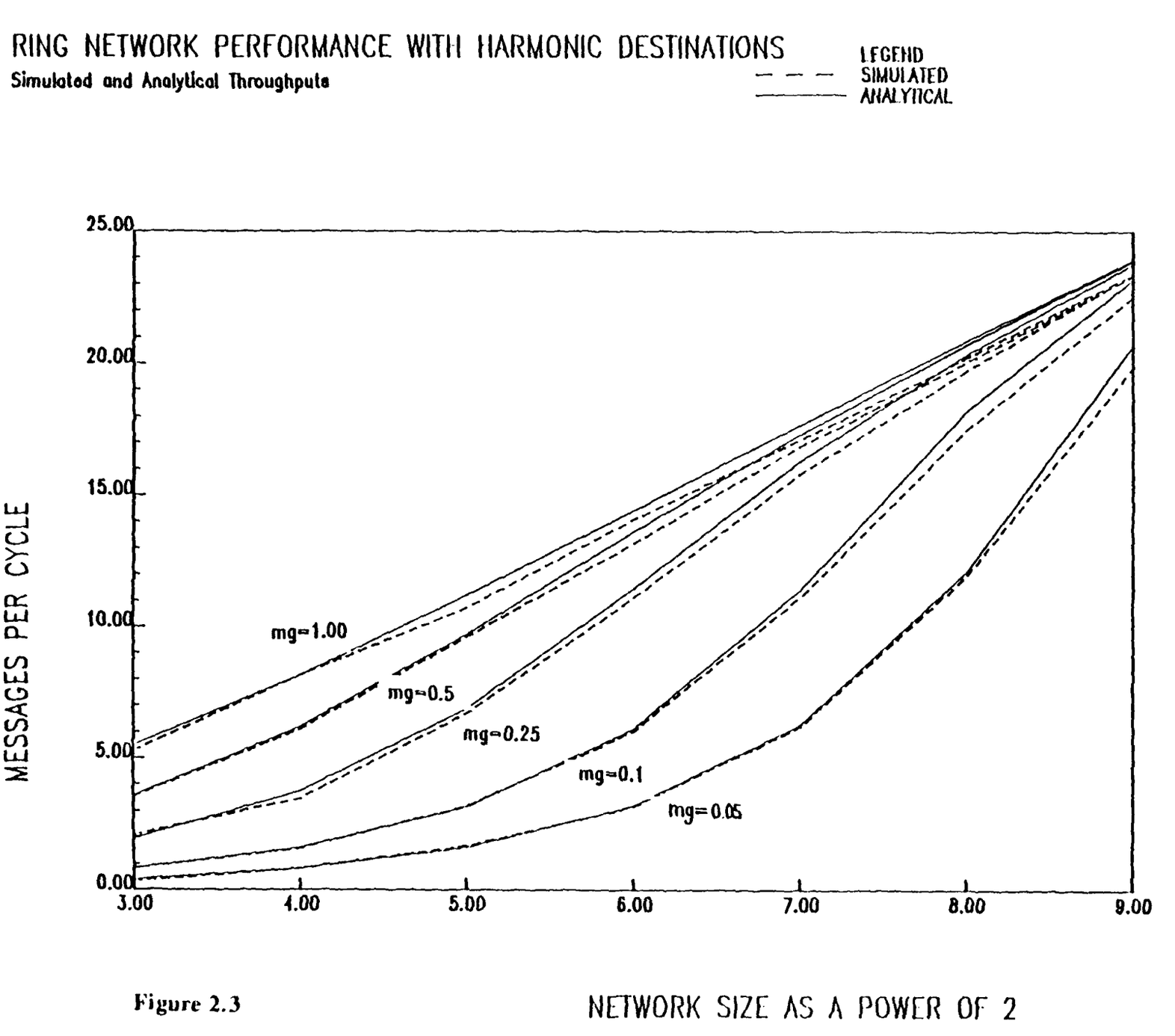

RING NETWORK PERFORMANCE WITH HARMONIC DESTINATIONS
Simulated and Analytical Throughputs
LEGEND
SIMULATED
ANALYTICAL
MESSAGES PER CYCLE
25.00
20.00
15.00
10.00
5.00
0.00
mg=1.00
mg=0.5
mg=0.25
mg=0.1
mg=0.05
3.00
4.00
5.00
6.00
7.00
8.00
9.00
NETWORK SIZE AS A POWER OF 2

Figure 2.3

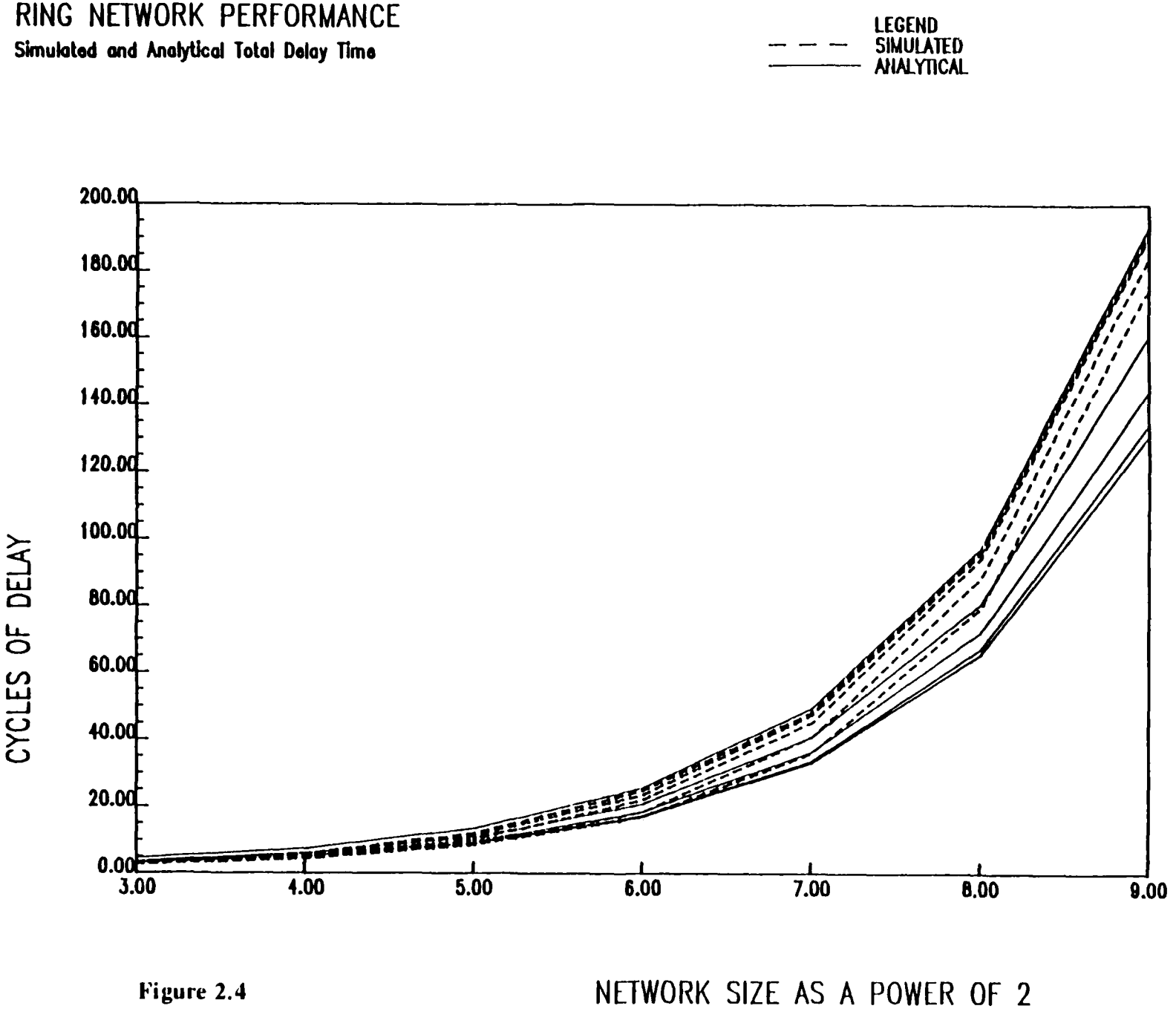

RING NETWORK PERFORMANCE
Simulated and Analytical Total Delay Time
LEGEND
SIMULATED
ANALYTICAL
CYCLES OF DELAY
200.00
180.00
160.00
140.00
120.00
100.00
80.00
60.00
40.00
20.00
0.00
3.00
4.00
5.00
6.00
7.00
8.00
9.00
NETWORK SIZE AS A POWER OF 2

Figure 2.4

$$P_t = \frac{\sum_{j=1}^{j=n} 1/j}{\sum_{i=1}^{i=n} \sum_{j=1}^{j=n} 1/j}$$

Neither the numerator and denominator can be expressed in closed form.

2.3. Throughput of ring networks

The throughput is defined as the average number of messages delivered at the end of each cycle or shift of the ring. Since m is the rate at which messages arrive at a node via the network, the number of messages delivered at a node is mP_t. Since there are N nodes in the network, the total throughput, denoted by T, can be expressed as follows.

For bidirectional ring $T=2mP_tN=8m(N-1)/N$

For unidirectional ring $T=mNP_t=2m$

In case of the locality, the throughput expression for the bidirectional ring cannot be expressed in closed form. It can be seen that the throughput for no locality asymptotically levels off to 2 and 8 for the unidirectional and bidirectional rings, respectively. Thus, the throughput of unidirectional rings can be quadrupled at only twice the cost.

Figures 2.2 and 2.3 graphically depict the throughput for bidirectional rings when messages have random and local destinations. Figure 2.4 shows the near-exponential increase in the total delay time (the theoretical derivations are not included here) which is composed of the waiting time in the buffer and the transit time over the network. As can be seen, the analytical results agree closely with the simulated results. It is of interest to note that when messages exhibit locality, the throughput reaches as much as 27. This is more than three times the throughput of rings with no local messages.

2.4. Conclusions

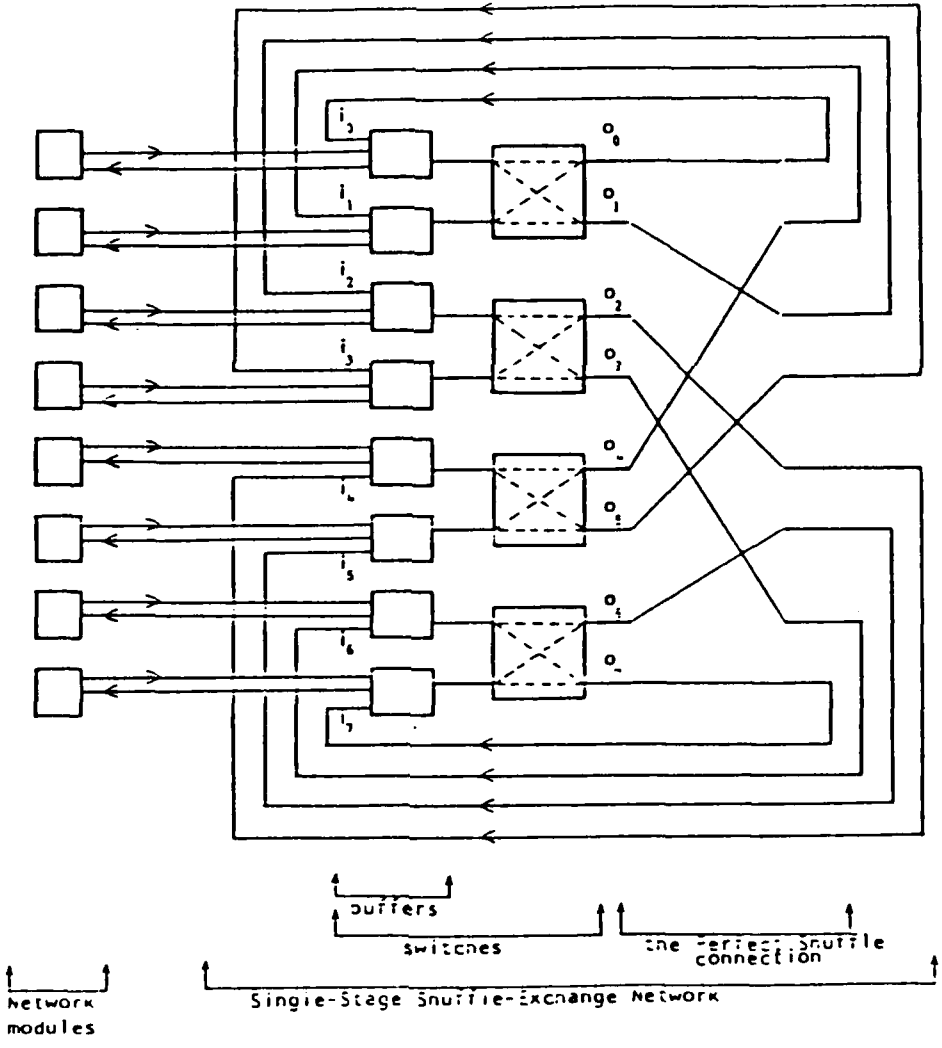
We have presented an analytical model to evaluate the throughput of ring interconnection networks in message-passing environments. Although the model is relatively simple, it effectively shows the serious limitations of the register-based ring networks. Clearly, from Figures 2.2. and 2.3, for processors using message passing for communications, the ring network cannot provide an acceptable throughput for more than about 16 processors.

We next evaluate other network topologies as possible candidates interconnecting the processors in SPARO.

3. PERFORMANCE OF SINGLE-STAGE AND REPLICATED SHUFFLE-EXCHANGE NETWORKS

The first alternative topology that we examined in detail is the shuffle-exchange network. We also examined the potential of employing replicated shuffle-exhange networks which have been used in electronic network designs to improve the performance

Figure 3.1 A shuffle-exchange network for 8 processors



of the single-stage network. Although analytical models for predicting the performance of these networks have been studied, the question of what the desired level of replication should be, has been left unanswered. We therefore examine, from an architectural perspective, which of the following possible networks is desirable:

i) a single shuffle-exchange network (SEN),

ii) a full multistage interconnection network consisting of $\log_2 N$ stages when N processor nodes are to be interconnected, or

iii) replicated SENs where the degree of replication is between 1 and $\log_2 N$.

3.1. The shuffle-exchange network

We will initially consider the single stage SEN. The operation of the SEN that we are considering is described in detail by Lang in [6]. A one-stage SEN contains $N=2^n$ registers or buffers, indexed from 0 to N−1, when N processors are connected to the network. Figure 3.1 shows the SEN for connecting 8 processors or network modules. To deliver messages from the processors, the network operates by cycles. In each cycle, a message and its destination tag (binary address of the destination processor) passes through an exchange element and then undergoes a shuffle permutation. If the destination processor is reached, the message is delivered to the processor, or else it is injected back into the network. It takes at most n periods for a message to reach its destination.

The shuffle stage of the SEN is used to realize a perfect shuffle among the N elements. The perfect shuffle can be described by the following relation:

$$S(i)=(2i+\lfloor 2i/N \rfloor) \bmod N$$

where S(i) denotes the destination of the message from the ith processor due the shuffle permutation. Figure 3.1 can be used to verify the shuffle stage for N=8. The second part of the SEN consists of N/2 exchange elements. The purpose of the exchange element is to exchange the destinations of messages arriving from two adjacent processors. The exchange is done based on the value of a control bit. Depending on the control bit, the exchange element will either exchange the position of the input messages or leave them unchanged. Instead of using separate control bits for routing a message through the SEN, one can conveniently use the destination tag method [10] for setting the control of the exchange elements. In this method, if $b_n b_{n-1} \ldots b_1$ be the binary representation of the destination processor, then during the $(i-1)$th period, bit b_i is used to set the exchange element. Depending on the controlling bit, then the switch will either exchange, i.e., cross-connect the input and the output, message packets or let them pass through undisturbed. Clearly, since each input message can provide the switch setting independently, there is a fifty-fifty chance of conflict when two messages arrive at an exchange element.

3.2. Operational model of the SEN and its analysis

A good operational model and its analysis has been presented in [5]. Here we will give a brief description of the model to motivate the study of replicated SENs.

In the normal operational mode, several messages will be circulating in the network. Both from an analysis as well as from an implementation viewpoint, it is easier to consider the synchronous operation of the network. Synchronous operation of the network implies that the exchange elements and the registers (that hold the messages to be injected to the network) are latched simultaneously. On the average, messages in a

SEN can be delivered within 2n cycles. There is no upper bound on the number of cycles required since in each pass through the network a message may get blocked, due to possible conflicts arising at the exchange elements of the network. The exchange element resolves this conflict by allowing one message to go through to the proper destination. If messages cannot be dropped, then the message that loses in the conflict resolution is routed via a longer path, thus increasing the 'delay time' or the number of cycles required to deliver it. Two schemes for resolving conflicts are often used:

i) random selection, and

ii) closest first selection.

In the random selection scheme, as the name implies, the message chosen for routing to the proper destination is chosen randomly. The message that loses in the random selection starts afresh in the routing cycle. To represent the status of a message in the routing, a counter is associated with the message. The counter is initially set to one. If the message is successfully routed in one pass or period of the SEN, its counter is incremented. However, if the message loses during a conflict resolution, its counter is reset. Thus, a message reaches its destination when the counter value is n+1. In the prioritized case of the closest first selection, the conflict is resolved by selecting the message with the larger counter value (randomly, if there is tie). Results from [5] show that, as expected, the message delay is smaller and the throughput is higher (for small loads) in the second case. Therefore, we have focussed our attention on SENs utilizing the prioritized conflict resolution scheme.

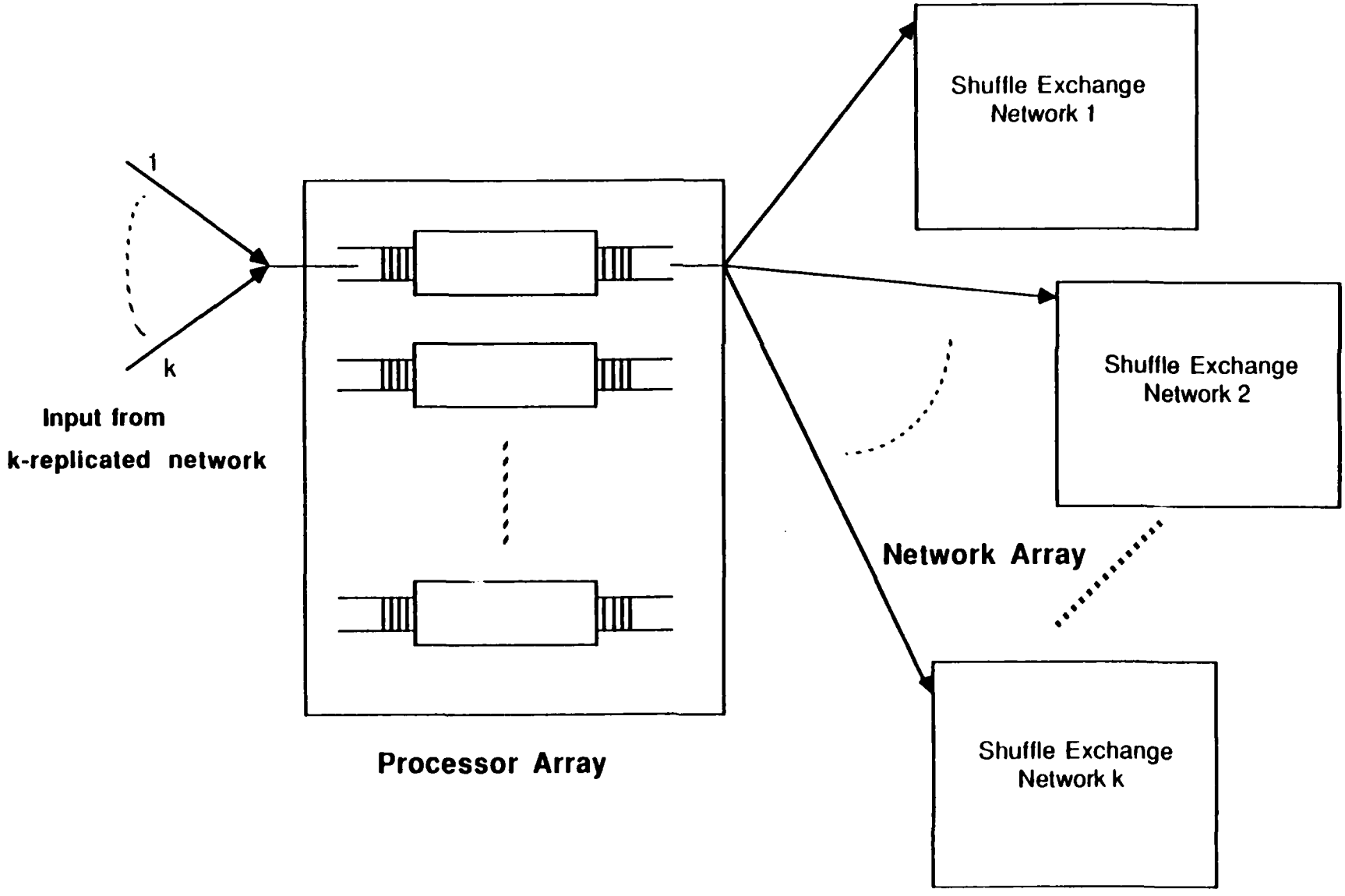
3.2.1. Previous analysis and results

Previous analysis of the SEN considered the state transition of the counter to determine the probability that a message has been delivered (i.e., the probability that the message has a counter value of n+1). This would then yield both the throughput and the expected number of periods (delay) that a message stays in the network. Unfortunately, the authors could not find a closed form expressions for these metrics for the prioritized selection case. Instead, numerical solutions have been provided for different 'loads', where the load is defined as the fraction of active messages to the total number (N) of processors. Note that the load is distinct from the rate of message generation by the processors. The results showed that for loads below 0.25, the expected delay is $1.5\log_2 N$ periods. For larger loads, the expected delay rises while the throughput falls off, especially for large N. In applying these established results to the general interconnection network problem that is of interest to us, we observe the following differences in operation.

First, the previous work considered connecting memory modules and processors and not processor-processor interconnection networks. Because the operating speeds of memory are lower than those of processors, the requests from processors are typically queued to the memory modules. In our case, the interconnection networks are used only to provide communication between processors. The network, as opposed to memory, will be the bottleneck. At high message generation rate, which results in high loads, the network will be fully loaded. In such a case we prefer to stop message generation rather than queue delivery requests. Thus, in our mode of operation, a message from a processor will be held in its buffer and not injected into the network if the network has to route a message through that processor-network path.

Second, the model assumes that all messages take n passes to get routed. This is the worst case routing time if no conflict occurs. The destination of a message can be

Figure 3.2 Replicated shuffle-exchange network



reached in fewer than n steps as an examination of the shuffle permutation will show. We will investigate this mode of operation where we will deliver a message when the processor address matches the destination tag, rather than when the counter reaches the value n+1. Such a delivery scheme is expected to give lower delay times than the results based on the assumption made in the analysis of [5]. For completeness, both delivery schemes will be examined.

Third and finally, the authors characterize the network by the load, which is the ratio of active messages to the total number of processors. In the case of processor interconnection networks, the characterization is usually in terms of the message generation rates of the processors. In general, as our results indicate, modest message generation rates, even below 0.5, can produce a fully loaded network. In such cases instead of using a single SEN, larger multistage networks or replicated SEN networks should be used to spread the load among different network stages.

Previous researchers [5] have mentioned briefly the use of replicated SENs for improved throughput when the load is high. A detailed study of the effectiveness of replicated networks has not been studied. We have chosen to examine the advantages of replicated SENs, in the operational mode described above, in greater detail.

3.3. Replicated Shuffle-Exchange Networks

Figure 3.2 illustrates what a replicated SEN (RSEN) looks like. If a k-replicated network is used, then k networks are connected in parallel. A message to be delivered can be routed to any available network. Similarly, at most k routed messages can arrive at the input of a processor. For each of the k networks, the effective number of processors generating messages is at most N/k. Thus, the expected load in each SEN in the k-replicated network is at most 1/k. In our study of RSENs, we have limited k to be less than or equal to n since any message takes at most n passes to be routed if no conflict occurs.

Our focus in the study of replicated networks is on the performance of k-replicated SENs for different values of k and for different message generation rates. The advantage in using RSENs, besides the increased throughputs, is the flexibility of varying the amount of replication. As Figure 3.4 shows the level of replication can be increased from 1 to 10 for a 1024 network for increased throughput. The cost and the desired performance will decide what level of replication is most suitable.

In the next section, we present the results on the performance of single SEN with dfferent network size, and the performance of k-replicated networks for different k.

3.4. Performance of single and replicated SENs

Two different sets of results have been obtained by simulation. First, we have examined the effect of the size of the network on the throughput for a fixed message generation rate. This has been done for a single SEN for the purposes of studying the effect of size on performance. Second, we have considered the effect of the degree of replication on the both the throughput and the delay time.

The performance of the single-stage SEN and the RSEN is given graphically in Figures 3.3 and 3.4. Figure 3.3 depicts the throughput and delay times for delivery of messages in a single-stage SEN of various sizes upto 1024. The message generation considered is only 0.25 since rates higher than this leads to a fully loaded network. A fully loaded network exhibits a very large number of conflicts resulting in very few deliveries per cycle. According to our simulations, a 1024 network has a throughput of only 40

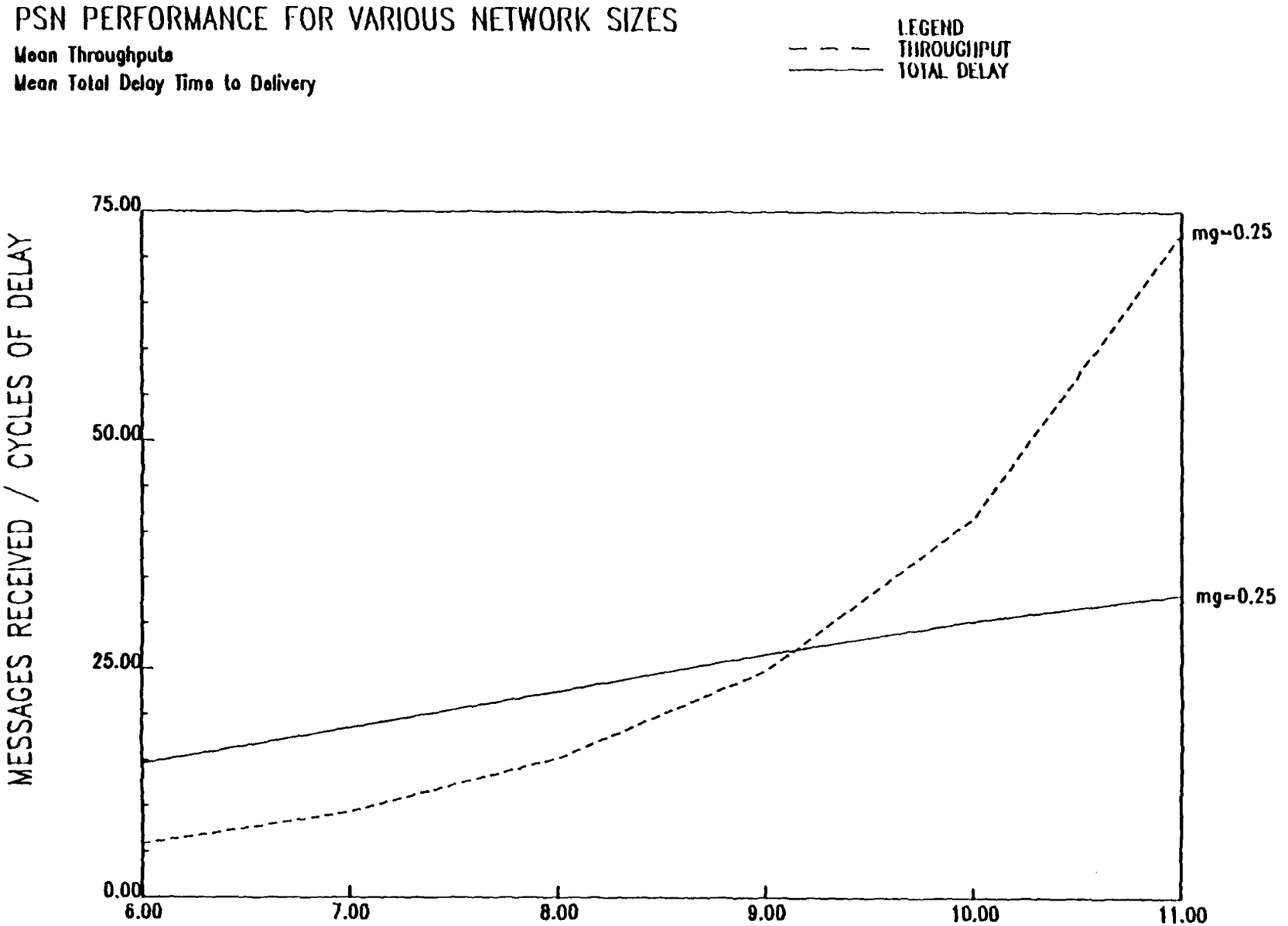

PSN PERFORMANCE FOR VARIOUS NETWORK SIZES
Mean Throughputs
Mean Total Delay Time to Delivery
LEGEND
THROUGHPUT
TOTAL DELAY
MESSAGES RECEIVED / CYCLES OF DELAY
75.00
50.00
25.00
0.00
mg=0.25
mg=0.25
6.00
7.00
8.00
9.00
10.00
11.00
NETWORK SIZE AS POWERS OF 2

Figure 3.3

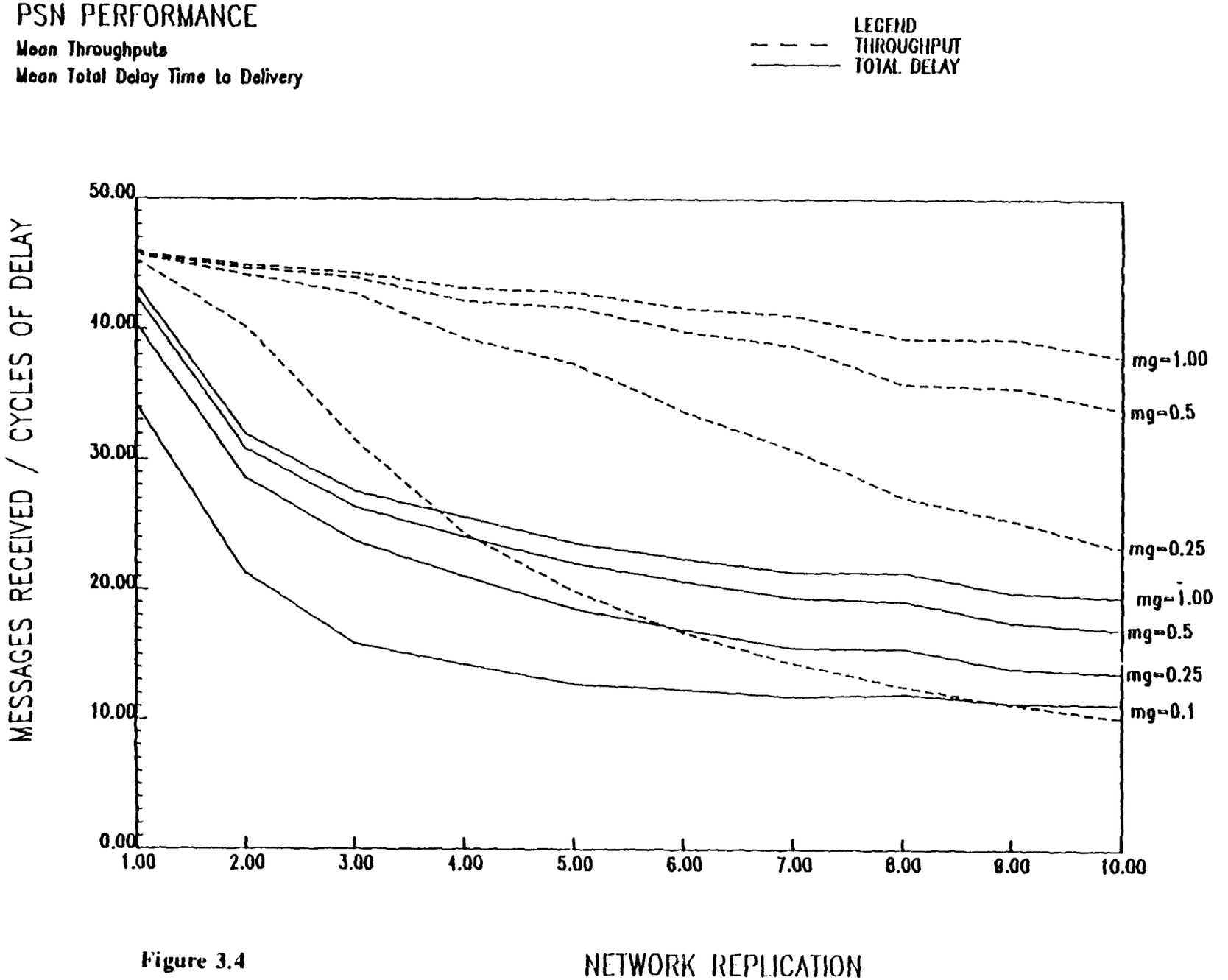

PSN PERFORMANCE
Mean Throughputs
Mean Total Delay Time to Delivery
LEGEND
THROUGHPUT
TOTAL DELAY
MESSAGES RECEIVED / CYCLES OF DELAY
50.00
40.00
30.00
20.00
10.00
0.00
1.00
2.00
3.00
4.00
5.00
6.00
7.00
8.00
9.00
10.00
mg=1.00
mg=0.5
mg=0.25
mg=1.00
mg=0.5
mg=0.25
mg=0.1
NETWORK REPLICATION

Figure 3.4

when 256 processors can generate a message on average (message generation rate of .25). The RSEN performance (Figure 3.4) shows the throughput can be increased using replication. Note that the throughput shown in Figure 3.4 is the throughput per individaul network. Thus, the total throughput for a 10-replicated network is greater than 380 when the message generation rate is 1.0. The dramatic increase in throughput in RSENs is due to the decreased loading in each network which results in fewer conflicts than in the single-stage SEN.

Another result obtained from simulations which influences the implementation is the effect of the delivery schedule of messages. In the normal delivery scheme, a message is delivered to its destination when the counter associated with the message reads $\log_2 N$. We had expected that delivering messages on the basis of the comparison of the destination address with the address of the node at the end of each cycle would be more efficient. However, simulations show there is no perceptible difference in the total delay time of messages or the throughput when the second delivery scheme is adopted. The lack of difference can be attributed to the effect of conflicts that erode any advantages expected in the scheme using address comparisons.

3.4.1. Performance of buffered SENs

It is worth noting here that we have also examined the issue of buffered single stage SENs. We examined the effect of first-in-first-out (FIFO) buffers to queue arriving messages being injected in the network. Interestingly, the performance of buffered SENs, even for modest loads, degrades drastically as the number of buffers increases. This is because message delays increases almost exponentially while the throughput remains almost constant once the network is fully loaded. While the results may not be obvious at first, the results can be explained as follows. When buffers are used, processors can generate messages while buffers are not full even when the network routes messages through them. At some point, when all buffers are non-empty, the network essentially has a load of 1. From earlier results [5], we know that the performance degrades as the load on the network is increased above 0.25.

Another problem in buffered single stage SENs is the problem of routing messages into a processor whose buffer is full. One approach [9] is to provide handshaking capability between source and destination processors. However, in a single-stage SEN, a chain of upto n handshakes maybe required (as in the multistage SEN [9]). When a buffer is full and a message has to be accomodated in the input queue, some message in the buffer has to be removed to some other processor. Such a scheme will require more complex control and therefore a buffered SEN does appear attractive.

To examine the performance of SENs and RSENs more objectively, we examine other candidate networks under similar conditions of size and load.

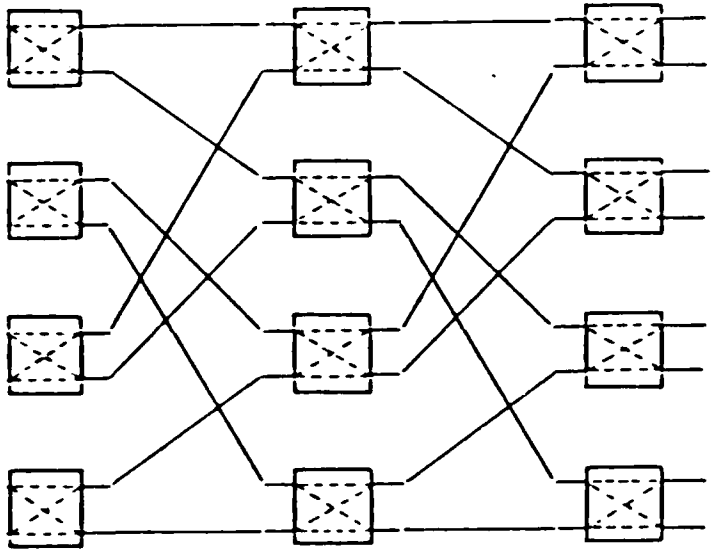
4. COMPARISON OF RSEN WITH OTHER NETWORKS

The strength of RSENs can be judged best on the basis of its performance relative to other well-known networks. We have therefore examined a number of interconnection networks that are commonly used in electronic parallel processing architectures.

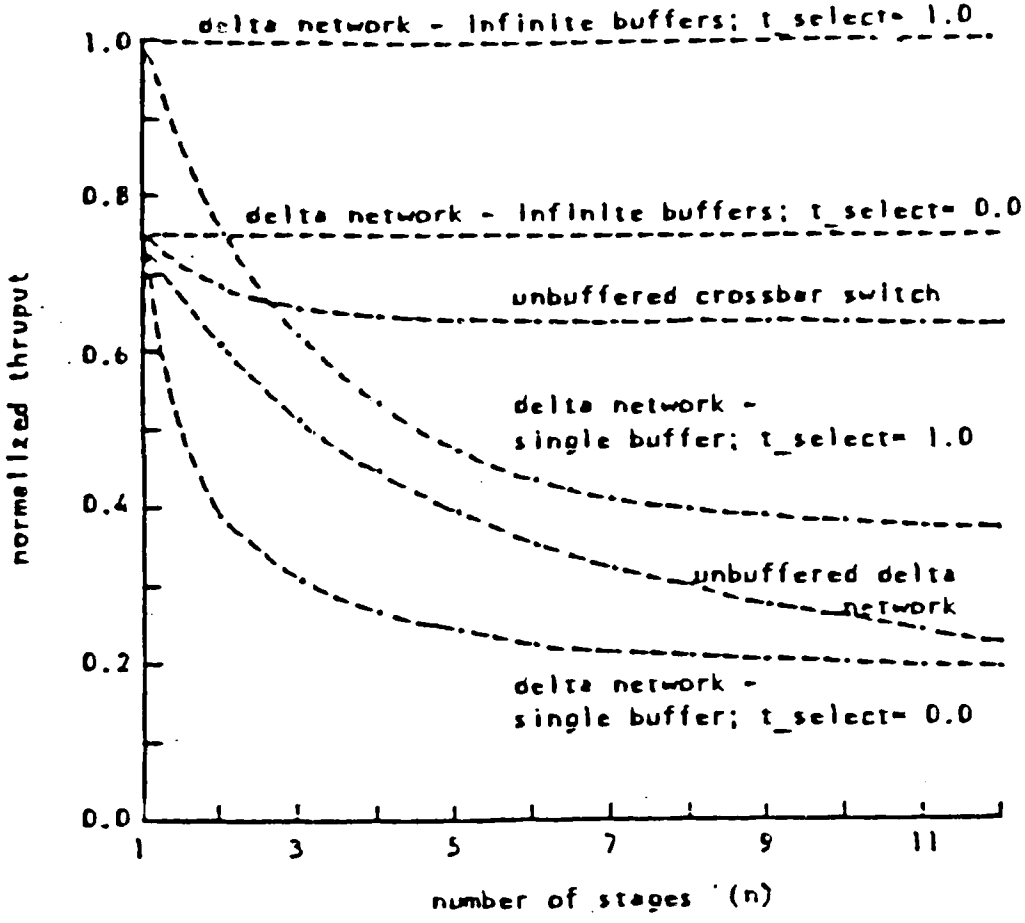
4.1. Comparisons with multistage interconnection networks

Multistage interconnection networks (MINs) are very popular in implementing large parallel processors. An example of such a network in a commercial machine is the BBN

Figure 4.1 Multistage Interconnection Network and its performance



Example of an 8 X 8 MIN.



Normalized throughput versus number of stages (n).

Butterfly. Figure 4.1 shows the topology for an 8 × 8 MIN. To compare the RSEN with the MIN, we have relied on the results on unbuffered delta networks (one class of MINs) given by Patel [8] and on the results on buffered delta networks by Dias and Jump [9]. Although both sets of figures, presented in Figure 4.1, are obtained from analytical models, Dias and Jump have verified their results by simulation.

The results for a MIN show that for a 1024 10-stage MIN, the normailized throughput (that is, the ratio of the absolute throughput to the total number of processors) is about 0.2 for an unbuffered network and 0.39 for a network with a single buffer. These figures translate to a throughput of 205 in case of the unbuffered network and a throughput of approximately 400 for a network with a single buffer. This is comparable to the throughput of a RSEN (380) composed of 10 networks. The normalized delay time to deliver a message in the MIN is about 1.5 or 15 cycles for a 10-stage network. This also compares well with the 15 - 20 cycle range observed in the RSEN.

In comparing the RSEN with the MIN, note that while the MIN has multiple ($\log_2 N$) switching stages, the processors in a k-replicated RSEN must be able to accept up to k messages on its input port. However, in the case of a RSEN there is an added flexibility of using less than $\log_2 N$ shuffle-exchange networks if less than maximum throughput is acceptable.

4.2. Comparisons with the hypercube

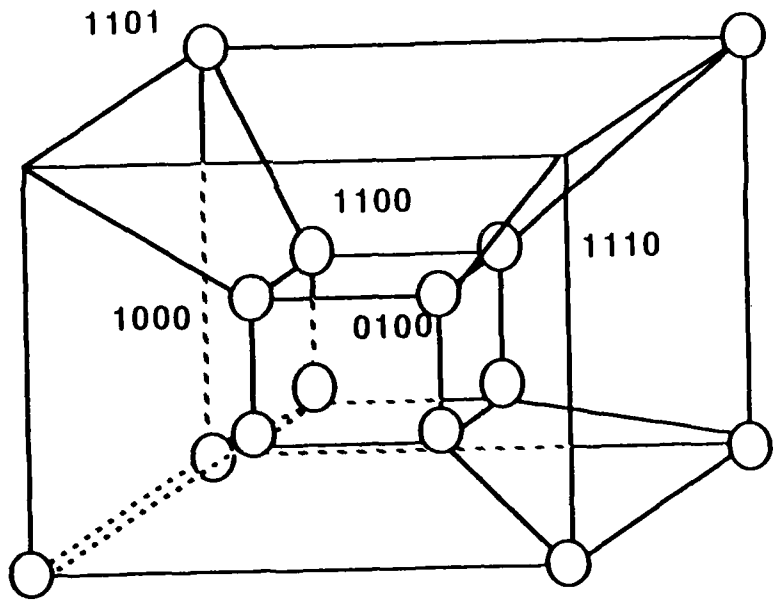
To compare the performance of RSENs with another popular interconnection network of comparable size, we have examined the hypercube topology. The hypercube has recently become popular by making its appearance in two commercial machines the Connection Machine (CM) [11] and the Intel Hypercube. Figure 4.2 shows the topology of a hypercube of 4 dimensions. In our analyses, we have considered a 10-dimensional hypercube.

As in the CM, each node in the hypercube is assumed to possess a routing buffer of length k ($0 \leq k \leq \log_2 N = n$). The routing in the hypercube is determined by forming the bit-by-bit EXOR of the source and destination addresses. The bit positions of the result indicate the dimensions along which routing takes place. The network operates synchronously with one dimension being active at a time. There is no set transfer sequence amongst the dimensions for a message to be routed.

When two processors are connected during some dimension cycle, each processor examines its own buffer to check for messages that need to be transferred. If none are found, a processor checks to determine if its buffer is full. If both processors have messages to transfer, a two-way transfer takes place. If only a single message packet needs to be transferred, the transfer is possible only if the buffer of the destination processor is not full. Two modes of operation are possible when considering the delivery of messages. In the first, both message generation and delivery are allowed in every dimension cycle. In the second, there is an upper bound on the number of messages that can be generated and delivered in every n dimension cycles or one network cycle. The second mode of operation is followed in the CM.

In the first mode of operation, since messages can be delivered in each dimension cycle, one expects a smaller waiting and therefore a smaller delay time. While this should result in better performance, the control is expected to be more complex and the dimension cycle would be longer than in the CM mode. The network cycles in the two modes of operation of the hypercube therefore have different meanings. In the first mode where deliveries are allowed in each dimension cycle, an individual dimension

Figure 4.2 Hypercube of 4 dimensions



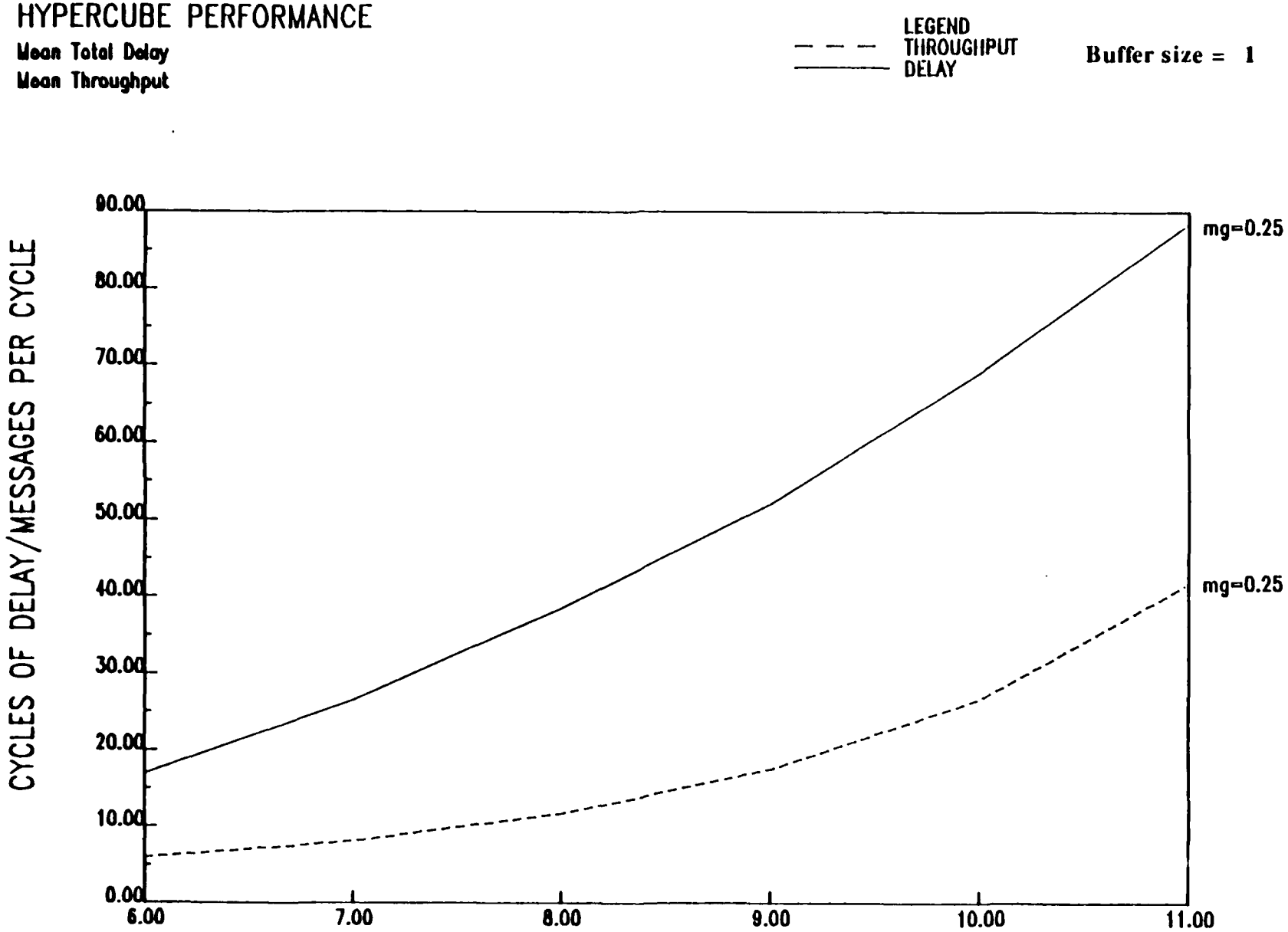

HYPERCUBE PERFORMANCE
Mean Total Delay
Mean Throughput
LEGEND
THROUGHPUT
DELAY
Buffer size = 1
CYCLES OF DELAY/MESSAGES PER CYCLE
90.00
80.00
70.00
60.00
50.00
40.00
30.00
20.00
10.00
0.00
6.00
7.00
8.00
9.00
10.00
11.00
mg=0.25
mg=0.25
NETWORK SIZE AS A POWER OF 2

Figure 4.3

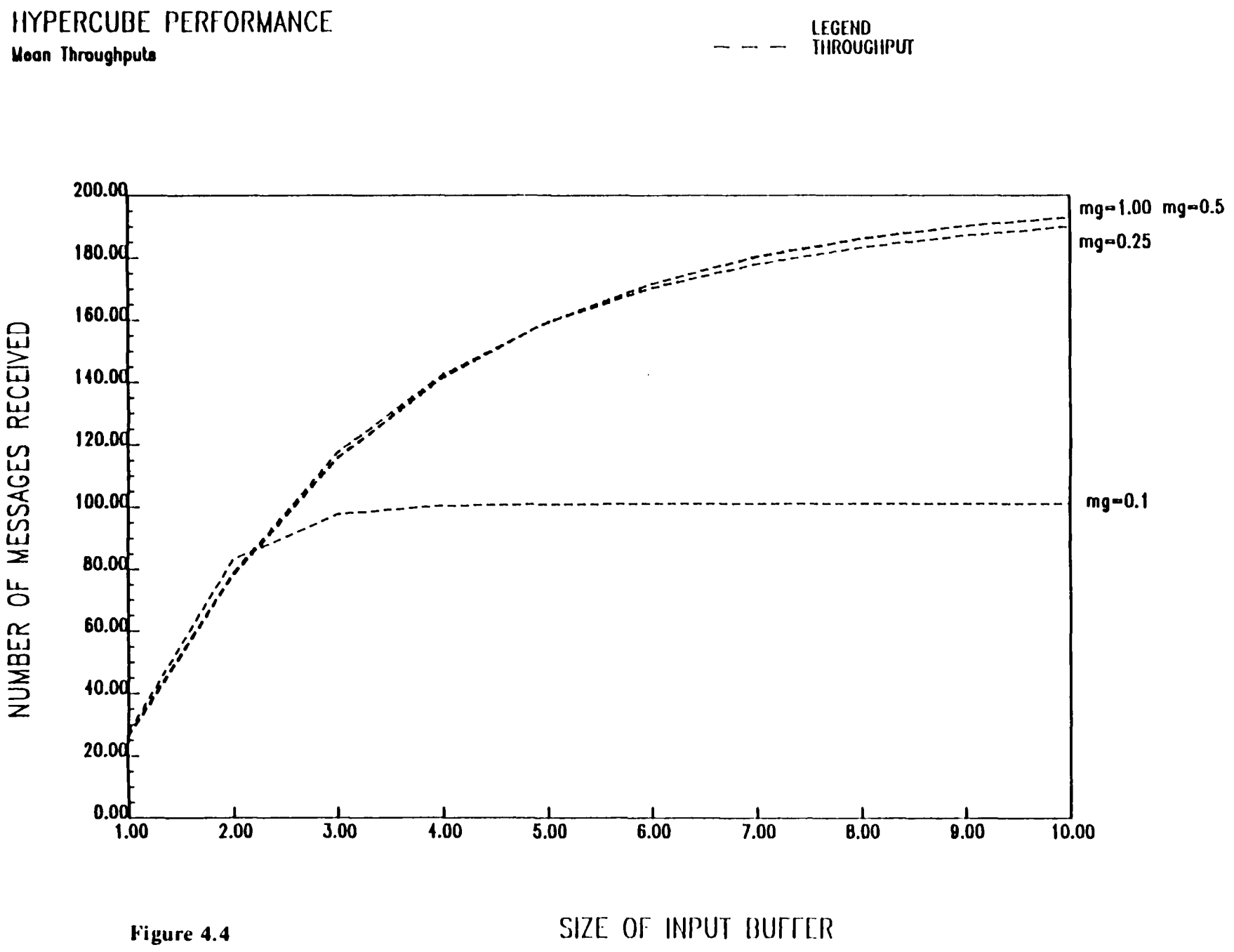

HYPERCUBE PERFORMANCE
Mean Throughputs
LEGEND
THROUGHPUT
NUMBER OF MESSAGES RECEIVED
200.00
180.00
160.00
140.00
120.00
100.00
80.00
60.00
40.00
20.00
0.00
1.00
2.00
3.00
4.00
5.00
6.00
7.00
8.00
9.00
10.00
mg=1.00 mg=0.5
mg=0.25
mg=0.1
SIZE OF INPUT BUFFER

Figure 4.4

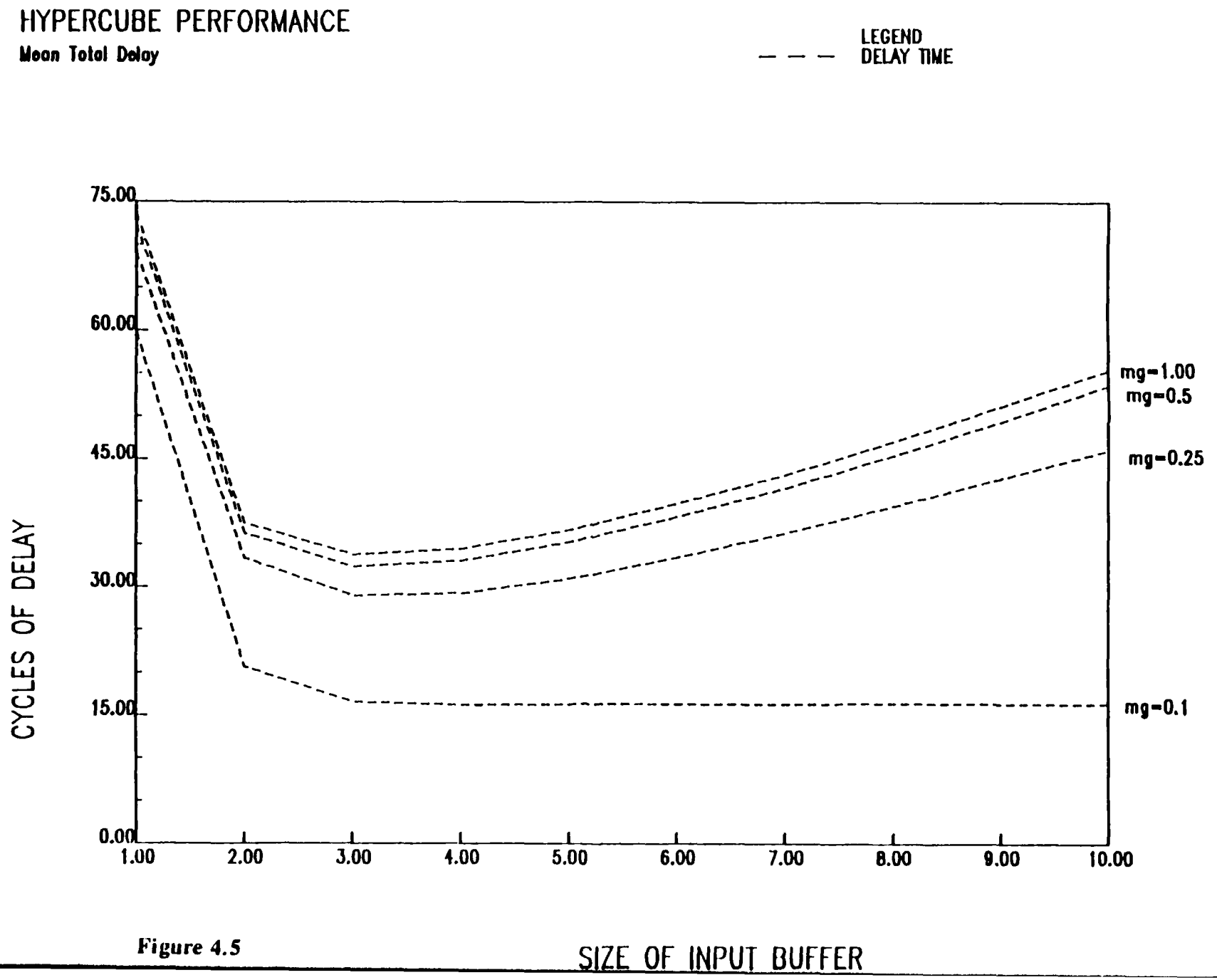

HYPERCUBE PERFORMANCE
Mean Total Delay
LEGEND
DELAY TIME
75.00
60.00
45.00
30.00
15.00
0.00
CYCLES OF DELAY
1.00
2.00
3.00
4.00
5.00
6.00
7.00
8.00
9.00
10.00
SIZE OF INPUT BUFFER
mg=1.00
mg=0.5
mg=0.25
mg=0.1

Figure 4.5

Figure 4.6 Hypercube operated in the Connection Machine mode

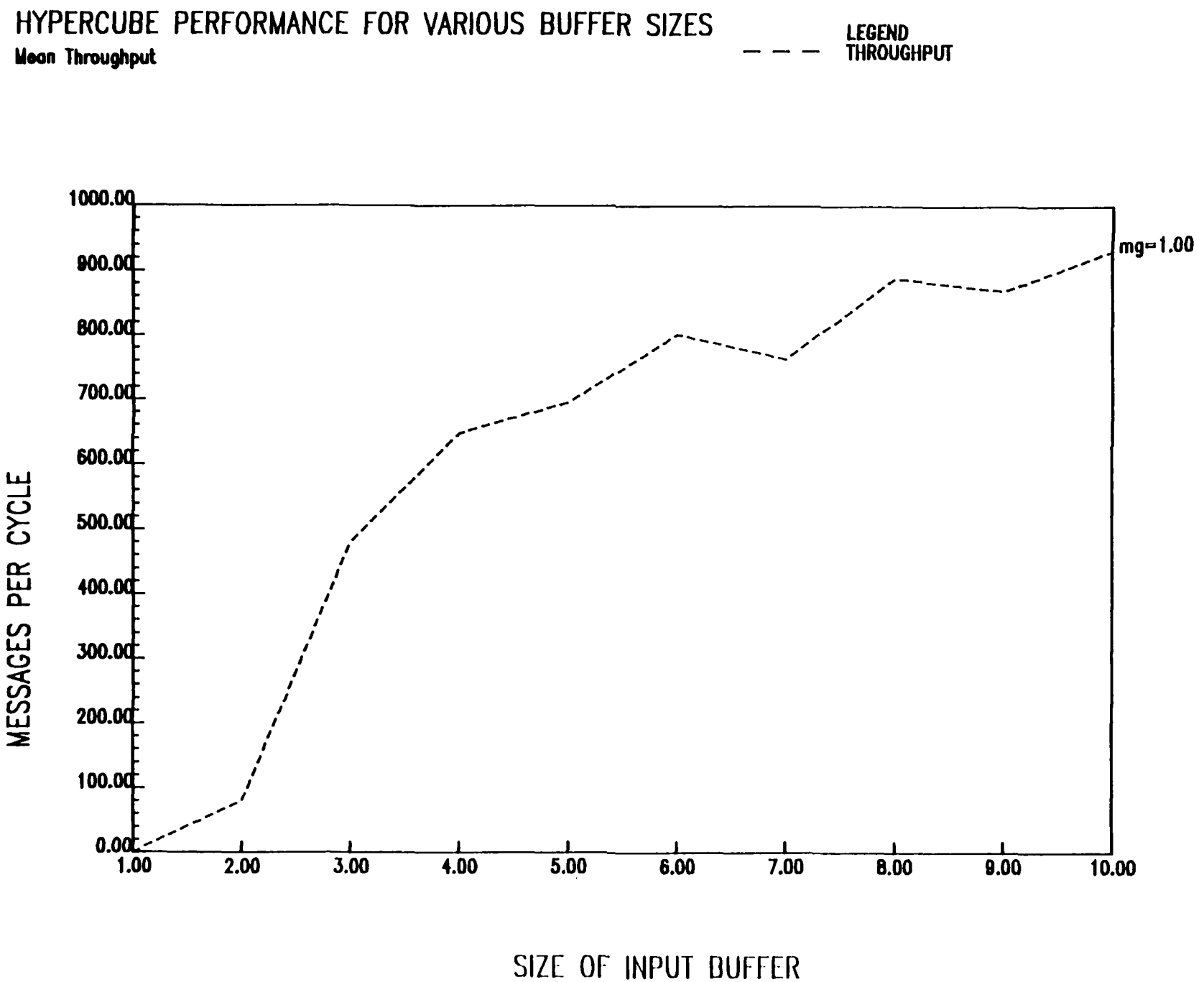
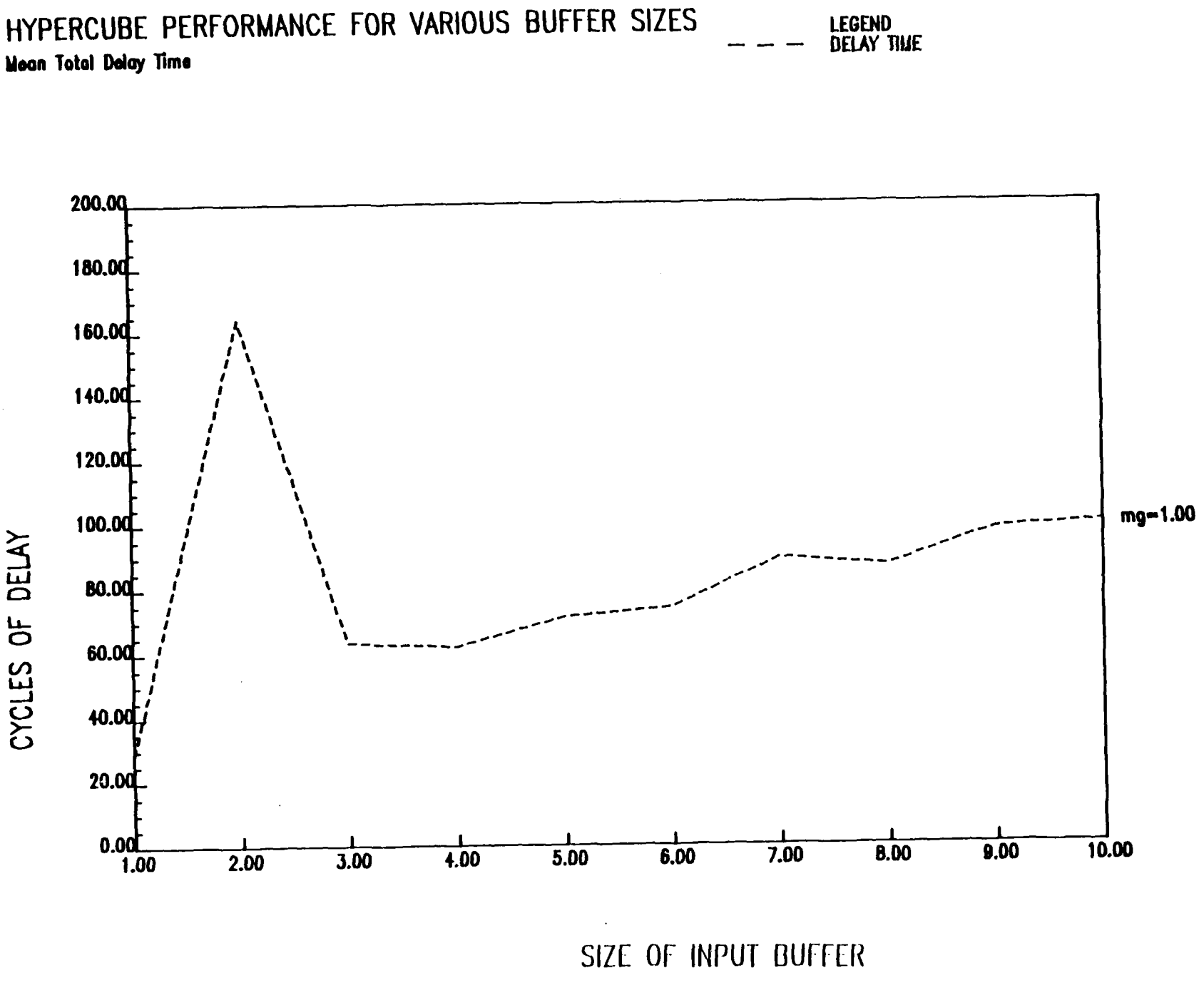


Figure 4.7 Hypercube operated in the Connection Machine mode



cycle is longer to allow for message deliveries. In the CM mode of operation, the network cycle is composed of simpler dimension cycles during which messages can only be transferred. All deliveries take place at the end of the network cycle. While we focussed our attention on the first method, we simulated both modes of operation for comparison.

Figure 4.3 shows how the throughput varies with the size of the hypercube when a single buffer is used. With a message generation rate of 0.25, the throughput is only about 25 for a hypercube of 10 dimensions. The poor throughput results from the overflow of the input buffer in each dimension cycle when more than one message have to be transferred across pairs of processors. Figure 4.4 examines the throughput for a 1024 hypercube as a function of the buffer size. The maximum throughput for such a network is about 190 when a buffer of size 10 is used. The total delay time is given in Figure 4.5 which shows that the delay time is minimized for a bufffer size of 3. For larger buffers the waiting time increases enough to deteriorate the total delay time.

In the CM mode of operation, where 1 message generation and 1 message delivery was allowed (that is, we allow 1 message to be generated, if the buffer is not full, and 1 delivery, if any message was generated, at the end of each network cycle), the throughput reaches a maximum value of about 1000 per network cycle. (Figures 4.6 and 4.7 show the throughput and delay time for the CM mode of operation.) In either mode of operation of the hypercube (a maximum of 200 per dimension cycle), we believe that the RSEN is very competitive (385 per single cycle). It must be noted that it is difficult to make an exact quantitative comparison between the two networks since the RSEN cycle of operation is different from the dimension cycle in the hypercube. As pointed out earlier, the dimension cycle in the CM mode of operation is shorter since no message deliveries until a network cycle. Actual implementation issues must be considered before accurate comparisons can be made.

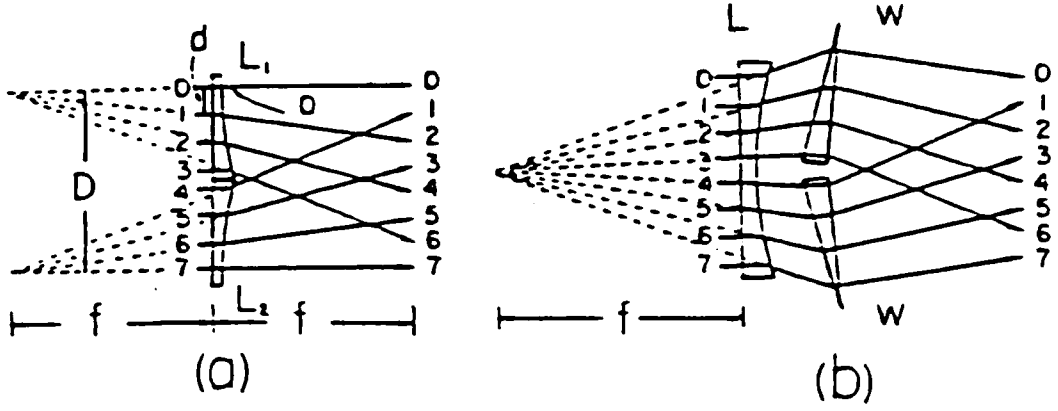
In terms of optical implementation, the SEN appears more attractive than the hypercube which must use large buffers for each node in the processor. On such simple first order analysis, the SEN cycle would be expected to be shorter than that of the hypercube. Thus, given that the two networks are competitive on the basis of throughputs, the RSEN would appear to be a better candidate.

5. IMPLEMENTATION OF OPTICAL RSENS

Our analysis of interconnection networks, based mostly on performance, reveals the RSEN to be very competitive to other commonly used topologies. The RSEN also appears more attractive than other networks because of recent work on the optical implementation of the perfect shuffle [12 - 15]. Lohmann [12] and Midwinter [15] initially showed how the perfect shuffle can be implemented very effectively using passive optical elements such as lens and prism combinations or holograms. Eichmann and Li [14] have later shown an even more compact implementation in optics which reduces the total optical path length from Lohmann's approach by a factor of 6. Their results indicate that the channel spacing d and the spot size α are the limiting factors.

$$d = \frac{D}{N-1}$$

$$\alpha \leq \frac{D}{2(N-1)}$$



Schematic diagrams of an OPS where D is the system aperture, a is the input channel size, and d is the channel spacing. (a) A lens-based system: L_1 and L_2 are two identical focal length (f) negative cylindrical lenses. (b) An alternative lens-based system where a single negative cylindrical lens together with two identical prism wedges are used.

Figure 5.1 Optical Perfect Shuffle (after Eichmann and Li)

where D is the aperture of the lens used (Figure 5.1). With a 50 × 50 sq. mm aperture lens, the optical perfect shuffle can handle as many as 40,000 light channels. The channel spacing and spot size is assumed to be 0.25 and 0.1 mm, respectively. While using bulk optics, as proposed in [14] may lead to larger-sized SENs, an alternative approach may be to use a holographic plate to accomplish the same shuffle permutation. The choice in implementing the optical shuffle will be pursued further in the next qurter.

The implementation of the complete shuffle-exchange requires using 2 × 2 switches for the exchange part. The switches can be realized using Wolloston or Rochon prisms with controllable halfwave plates [12, 13]. Unfortunately, the control in these switches is limited by electronic bandwidths although the perfect shuffle implementation can be operated at optical bandwidths. As mentioned by other researchers [15], embedding a packet switching network in a optical network for ultrafast routing requires considering additional infrastructures in optics/opto- electronics, especially for handling the collisions at the switches. We are in the process of identifying the critical functionalities required to implement the exchange stage of the complete SEN (and a RSEN). These are:

i) the switching control of the exchange elements,

ii) the control mechanism for conflict resolution, and

iii) the mechanism for delivering packets from the network to the processor, that is. how to redirect a message when it reaches its destination and not inject it back into the network.

To make the design process tractable, we have begun examining the requirements for a simple exchange element and issues involved in incorporating each of the three functionalities in an optical environment.

6. CONCLUSIONS

Despite the complex issues in implementing an optical SEN network, the major advantage of such an implementation is clearly in the tremendous increase in the overall message bandwidth. Besides the potential advantages of using optoelectronic switching in the exchange elements, there is also a great speed advantage in implementing the perfect shuffle compactly in optics. In case of an electronic implementation of the SEN, because of the non-modular layout involved, large fanout devices and long delay paths are expected to degrade the cycle time of the network. Typically, a 1024 or larger shuffle may require board-level interconnects. The bottleneck is expected to be not only in the switching element but also in the transfer of data electronically.

Our efforts in the previous quarter have revealed that the implementation of the SPARO graph reduction processor appears more feasible in high-speed GaAs rather than in opto-electronics. Using GaAs processors would still facilitate the building of the interface to the optical network. Since the bottleneck in the parallel reduction of combinator graphs is in the communication between the processors, a high-speed optical network can lead to much greater performance than obtainable in electronics.

REFERENCES

[1] M. Derstine, A. Guha, and S. Natarajan, 'A Conceptual Design of an Optical Symbolic Computer for Combinator Graph Reduction,' to be submitted to the Journal

of Applied Physics.

[2] S. Abraham and K. Padmanabhan, 'Performance of the Direct Binary n-Cube Network for Multiprocessors,' Proc. International Conference on Parallel Processing, 1986, pp. 636 - 639.

[3] T. Kushner, et al, 'Image Processing on the ZMOB,' IEEE Transactions on Computers, October 1982, pp. 943 - 951.

[4] R. Barry and K.M. Chandy, ' Performance Models of Token Ring Local Area Networks,' SIGMETRICS 1983, pp. 266 - 274.

[5] D.H. Lawrie and D.A. Padua, 'Analysis of Message Switching with Shuffle-Exchanges in Multiprocessors,' Proc. of the Workshop on Interconnection Networks for Parallel and Distributed Processing, 1980, pp. 116 - 123.

[6] T. Lang, 'Interconnections Between Processors and Memory Modules Using the Shuffle-Exchange Network,' IEEE Transactions on Computers, May 1976, pp. 496 - 503.

[7] C.P. Kruskal and M. Snir, 'The Performance of Multistage Interconnection Networks for Multiprocessors,' IEEE Transactions on Computers, December 1983, pp. 1091 - 1098.

[8] J.H. Patel, 'Performance of Processor-Memory Interconnections for Multiprocessors,' IEEE Transactions on Computers, October 1981, pp. 771 - 780.

[9] D.M. Dias and J.R. Jump, 'Analysis and Simulation of Buffered Delta Networks,' IEEE Transactions on Computers, April 1981, pp. 273 - 282,

[10] D.H. Lawrie, 'Access and Alignment of Data in an Array Processor,' IEEE Transactions on Computers, December 1975, pp. 1145 - 1155.

[11] W.D. Hillis, 'The Connection Machine,' MIT Press, 1985.

[12] A.W. Lohmann, 'What Classical Optics can do for the Digital Optical Computer,' and 'Optical Perfect Shuffle,' Applied Optics, 15 May 1986, Vol. 25, No. 10.

[13] C.W. Stirk, R.A. Athale, C.B. Friedlander, private communication.

[14] G. Eichmann and Yao Li, 'Compact Optical Generalized Perfect Shuffle,' Applied Optics, 1 April 1987, Vol. 26, No. 7.

[15] J.E. Midwinter, 'Novel Approach to the Design of Optically Activated Wideband Switching Matrices,' to appear in the IEE Proceedings, 1987; Pt. J, No. 6, December 1985.

[16] N. Weste and K. Eshraghian, Editors, 'Principles of CMOS VLSI Design,' Addison-Wesley, 1985.